Relatório EP2 - MAC0460

Lucas Quaresma Medina Lam

Nº USP: 11796399

1. Introdução

Neste projeto, vamos relatar os preparativos para uma expedição liderada pelo Professor Carvalho, que tem o objetivo de explorar uma região desconhecida. Vamos focar em encontrar e estudar pokémons dos tipos Aquático e Normal que ainda não foram registrados na Pokédex. Teremos apenas a ajuda do Pikachu para obter informações sobre os tipos dos pokémons desconhecidos.

Objetivo:

O principal objetivo deste relatório é criar um classificador binário para os pokémons dos tipos Aquático e Normal, levando em conta as restrições impostas pelas condições da expedição. Será desenvolvido um modelo que seja capaz de identificar se um pokémon desconhecido pertence a esses tipos, com base em suas características e em como eles se saem contra pokémons elétricos. Vamos usar a coluna "type1" para determinar a classe dos pokémons, e as colunas "type2" e "classification" serão descartadas, já que não teremos acesso a essas informações para os novos pokémons a serem descobertos.

2. Metodologia

a. Descrição do conjunto de dados:
O conjunto de dados contém informações sobre todos os 802 Pokémon de todas as Sete Gerações de Pokémon. As informações contidas neste conjunto de dados incluem estatísticas básicas, desempenho contra outros tipos, altura, peso, classificação, passos de ovo, pontos de experiência, habilidades etc. Ele pode ser obtivo pelo link https://www.kaggle.com/datasets/rounakbanik/pokemon

b. Modelos, parâmetros, hiperparâmetros e recursos utilizados:
Foram testados os seguintes modelos de classificadores:

- Regressão Logística
- Support Vector Machine
- Decision Tree
- Random Forest
- Gradient Boosting
- K-Nearest Neighbors
- Gaussian Naive Bayes
- AdaBoost
- Naive Bayes

Para cada modelo, foram ajustados alguns parâmetros e hiperparâmetros para otimizar o desempenho dos classificadores. Com os parâmetros iniciais como:

```
param_grid = {}
    if classifier_name == 'Logistic Regression':
        param_grid = {'classifier__C': [0.1, 1, 10, 100]}
    elif classifier_name == 'Decision Tree':
        param_grid = {'classifier__max_depth': [None, 3, 5, 7, 10]}
    elif classifier_name == 'Random Forest':
        param_grid = {'classifier__n_estimators': [10, 50, 100], 'classifier__max_depth': [None, 3, 5, 7, 10]}
    elif classifier_name == 'Support Vector Machine':
        param_grid = {'classifier__C': [0.1, 1, 10, 100], 'classifier__gamma': [0.1, 1, 'scale', 'auto']}
```

Utilizou-se o GridSearchCV para realizar uma busca exaustiva nos diferentes valores dos parâmetros e encontrar os melhores modelos, onde o resultado foi:

```
Classificador: Logistic Regression
Melhores hiperparâmetros: {'classifier__C': 0.1}

Classificador: Support Vector Machine
Melhores hiperparâmetros: {'classifier__C': 10, 'classifier__gamma': 1}

Classificador: Decision Tree
Melhores hiperparâmetros: {'classifier__max_depth': 5}

Classificador: Random Forest
Melhores hiperparâmetros: {'classifier__max_depth': 3, 'classifier__n_estimators': 100}

Classificador: Gradient Boosting
Melhores hiperparâmetros: {'classifier__learning_rate': 0.01, 'classifier__n_estimators': 50}

Classificador: K-Nearest Neighbors
Melhores hiperparâmetros: {'classifier__n_neighbors': 5}

Classificador: Gaussian Naive Bayes
Melhores hiperparâmetros: {}

Classificador: AdaBoost
Melhores hiperparâmetros: {'classifier__learning_rate': 0.1, 'classifier__n_estimators': 50}

Classificador: Naive Bayes
Melhores hiperparâmetros: {}
```

c. Avaliação dos modelos:
A avaliação dos modelos foi realizada utilizando a técnica de k-fold cross-validation com k=10. Essa tecnica divide o conjunto de dados em 10 partes, onde cada uma delas é usada como conjunto de teste uma vez, enquanto as outras partes são utilizadas para treinamento. Isso faz com que possamos ter uma estimativa mais robusta do desempenho dos modelos, evitando viéses de utilizar apenas um conjunto de treinamento e teste. Será mostrada a média do score do CV na seção resultados.

3. Resultados:

Os resultados foram avaliados utilizando a acurácia como métrica de desempenho. A acurácia mede a proporção de classificações corretas em relação ao total de amostras. Além disso, foram plotadas as matrizes de confusão para visualizar os resultados das classificações e entender melhor o desempenho dos modelos.

> 💡 *Acurácia dos modelos nos dados de treino e de teste, e suas respectivas matrizes de confusão; e o gráfico mostrando a relação das importâncias das características com base nos dados de teste utilizando a permutação estão como material suplementar no arquivo .ipynb*

4. Discussão:

a. Avaliação das características relevantes:
Para a seleção das características mais relevantes, foi utilizado um conhecimento prévio do problema (experiência em pokémon) onde foi selecioando algumas features que poderiam ser relevantes na escolha das características para treinamento dos modelos, sendo elas: "against_electric", "attack", "defense", "speed", "hp", "height_m" e "weight_kg".

Foi utilizada a função `permutation_importance` do scikit-learn para calcular a importância das características usando permutação. Em seguida, foi ordenada as características com base na importância média e selecionada as duas mais importantes. Com base nos resultados obtidos, as características mais relevantes para a classificação foram "against_electric", "attack", como apresentado no gráfico da seção de resultados. Vemos que “against_eletric” tem uma importância muito grande na classificação dos pokémons, sendo discrepante sua contribuição em relação aos outros atributos.

b. Utilidade dos resultados para o professor Carvalho:
Com uma acurácia média para todos os modelos de aproximadamente 87% e acurácia do melhor modelo em

aproximadamente 91% os resultados obtidos podem ser úteis para o professor Carvalho na classificação de novos pokémons encontrados durante a expedição com uma precisão satisfatória. Isso pode contribuir para a expansão da Pokédex e o conhecimento sobre a diversidade de pokémons.

Para melhorar a tarefa, poderiam ser utilizados outros pokémons além do pikachu, visto que o “agaist_*” revela dados importantíssimos para a tarefa de classificação de tipos.

5. Conclusão:

Este projeto demonstrou a viabilidade de desenvolver um classificador binário para identificar pokémons dos tipos Aquático e Normal com base em suas vantagens/desvantagens contra pokémons elétricos. Diferentes modelos de classificação foram testados e ajustados para obter os melhores resultados. Os resultados obtidos fornecem informações valiosas para o professor Carvalho na classificação de novos pokémons encontrados durante a expedição. Com melhorias adicionais, como a consideração de outros tipos de pokémons, esse classificador pode ser aprimorado e ampliado para auxiliar em futuras descobertas na Pokédex.

Referências:

- Abu-Mostafa, Y. S., Magdon-Ismail, M., & Lin, H. T. (2012). Learning from Data. AMLBook.
- Matplotlib. (2023). Matplotlib Documentation. Disponível em: **https://matplotlib.org/stable/index.html**. Acesso em: 4 jul. 2023.
- Pandas. (2023). Pandas Documentation. Disponível em: **https://pandas.pydata.org/docs/user_guide/index.html**. Acesso em: 4 jul. 2023.
- Raschka, S. (2019). Python Machine Learning. Packt Publishing.
- Scikit-learn. (2021). Supervised Learning. Disponível em: **https://scikit-learn.org/stable/supervised_learning.html#supervised-learning**. Acesso em: 4 de julho de 2023.
- Seaborn. (2023). Seaborn Documentation. Disponível em: **https://seaborn.pydata.org/**. Acesso em: 4 jul. 2023.